Chapter XI
Agile Development of Rule Systems

Joachim Baumeister
University of Würzburg, Germany

Dietmar Seipel
University of Würzburg, Germany

Frank Puppe
University of Würzburg, Germany

ABSTRACT

The engineering of rule-based systems was a relevant issue in the past decades and has become more attractive recently due to the emergence of intelligent applications on the web. For example, web application servers including rule engines and the rule-based description of (semantic) web services. This development allows for a natural formalization of business logic but also poses new challenges with respect to the acquisition and evolution of rule-based knowledge. Although, a lot of research has been done in the past, the practical engineering and life cycle of larger rule-based systems still remains to be a difficult and complex task. State-of-the-art tools for rule development provide extensive support for the engineering, the analysis and debugging of such systems, but a simple and easily adoptable methodology is missing. In the best case, such a methodology should be well understood even with little training and should provide techniques to ensure quality, evolutionary health, and cost control of a rule base project. We present an agile methodology for the development and evolution of rule-based systems. Intuitive concepts like the system metaphor, the planning game, and the implementation cycle make its adoption to arbitrary projects very easy. With the promotion of continuous techniques such as automated testing and refactoring we cope with evolutionary aspects of knowledge bases.

INTRODUCTION

Today, we see successful applications of rule bases in many domains, for example for the control of complex devices and business rules for the management of enterprise processes. The formalization using rules is one of the oldest knowledge representation methods.

Although different formalizations for knowledge representation have been proposed in the past, for example Bayesian networks, case-based reasoning, and consistency-based models, the use of rules is still popular and successful in many domains. When compared to alternative knowledge representations, the use of rules is often more intuitive for little trained users, since their semantics and inference is quite natural.

In recent years, rule engines have become popular in the commercial but also in the open-source world. Now, rules are practically applicable for everyone. However, although we see many successful applications using rules, the actual development of rule systems is still a complex and costly task. With the growth of a rule base the extension, modification, and refinement of the knowledge becomes more complex.

In the past, a collection of process models for the development and evolution of knowledge bases was proposed, for example, based on prototyping (Budde et al. 1992) or document-centric approaches like the comprehensive CommonKADS methodology (Schreiber et al. 2001). Experiences made in many knowledge system projects showed that such process models are difficult to apply in small-sized, innovative projects. Here, the estimation of development costs are a major concern and - due to the vague scope of the project - the flexibility during the development is also an important characteristic. Additionally, prototypical processes typically show only little support for the evolution of the developed knowledge and fail when the systems grow. For heavy-weight approaches like CommonKADS, the cost-effectiveness - the trade-off between development costs and utility - is difficult to estimate. Furthermore, documentation and design phases are preceding the actual implementation, although early feedback is a necessary requirement in many projects.

In this chapter, we introduce an agile process model for the engineering of rule systems. It was motivated by eXtreme Programming (XP), a well-known software engineering approach (Beck 1999). XP has gained much positive attention and proved its practical relevance in numerous projects. We adapted the most significant ideas of XP to define the agile knowledge engineering methodology. Like XP, we expect that using the process will yield early feedback during the project, a flexible and transparent decision process of system features, motivating „quick wins“ and a simple adaptation of the process to be applied by end-users, for example, the domain specialists.

In the following sections we describe the phases of the cyclic process including the definition of the system metaphor, the planning game, the implementation, and the final integration phase. Furthermore, we introduce appropriate techniques for testing and refactoring the evolving rule base. Both techniques are essential methods for the agile development of rule bases since they guarantee a safe and structured growth of a knowledge base. We also show how the agile process can be put into practice by using the knowledge modeling environment KnowME. Thereafter, related methodologies are discussed in the context of the agile process model.

THE AGILE PROCESS MODEL

In general, an agile process model tries to reduce the usual ballast of heavy-weight processes by proposing only a basic set of phases for the overall development process. The process model introduced here describes the cyclic execution of elementary phases that are described in more detail in the following.